

The Critical Role of Metadata in JSF Development

Roy Scrudder
Applied Research Laboratories
The University of Texas at Austin
P.O. Box 8029, Austin, TX 78713
512-835-3857
scrudder@arlut.utexas.edu

Dr. W. Henson Graves, Tom Teigen, Chris Johnson
Lockheed Martin Aeronautics Company, Joint Strike Fighter Program
Post Office Box 748, Fort Worth, TX 76101
817-777-1856, 817-378-2604, 817-777-1856
henson.graves@lmco.com, tom.s.teigen@lmco.com, john.c.johnson@lmco.com

Steve Hix
Paradigm Technologies, Inc.
1215 Jefferson Davis Hwy, Suite 1104, Arlington, VA 22202
703-418-4060
hix@pti-usa.com

James W. Hollenbach
Simulation Strategies, Inc.
8615 Conover Place, Alexandria, VA 22308-2515
703-360-3902
jimh@simstrat.com

Keywords:
Metadata, VV&A, JSF, SBA, DPD

ABSTRACT: *As acquisition program decisions are increasingly dependent on the results of modeling and simulation, the need for complete, trustworthy (authoritative) information as inputs to the modeling process becomes critical. The data required encompasses both the product under development and its external environment (e.g., threats, scenarios, weather). To judge data trustworthiness, one must know its derivation history and understand the conditions under which it is valid. This is data about data, or metadata. Thus determining trustworthiness requires careful management of the metadata about each instance dataset under consideration. The Joint Strike Fighter (JSF) Program is executing such an effort.*

A necessary step in managing product data is establishing configuration-managed, persistent storage. The data must also be made readily available to authorized users over a network and logical consistency must be assured. Each of these needs influences metadata requirements. Commercial product data management (PDM) systems routinely capture some metadata as part of the process of authoring and registering data. However, trustworthiness requires capturing more information than traditionally has been done. An example of this is the operational context in which system-interaction data is valid. JSF has established a comprehensive metadata specification for product and external environment data. This metadata specification is being used in the JSF Authoritative Modeling Information Systems (JAMIS).

This paper describes the JSF metadata requirements, the JSF metadata model and its implementation within JAMIS to support JSF product development, including system verification and test.

1. Introduction

This paper is the third in a series of papers presented at Simulation Interoperability Workshops addressing the challenges of managing information for an extremely large, extremely complex Simulation Based Acquisition (SBA) program. The first paper [1] outlined the general challenges of a coordinated effort between the United States Department of Defense (DoD) and the prime contractor to apply modeling and simulation (M&S) and manage the variety of information associated with M&S activities. The second paper [2] described the architecture of the information system that will be used to address these challenges. Although we will provide below a brief background on the Joint Strike Fighter (JSF) program, these challenges, and the information system architecture, the reader is encouraged to delve into the first two papers for a deeper background of the problems and solutions. In this paper, we will focus on the metadata (data about data) that enables managing the complexity, coherency, and consistency of the huge amount of M&S-related data required by the JSF enterprise.

The JSF Program will produce the affordable, next-generation strike aircraft weapon system for the U.S.

Navy, Air Force and Marines along with a number of international partners. Three variants will be built – conventional takeoff and landing, short takeoff and vertical landing, and carrier-capable. The JSF program has firmly embraced SBA principles [3-7]. At the time of this paper presentation, the JSF program will be nearly two years into the System Design and Demonstration (SDD) phase, with Lockheed Martin Aeronautics Company (LM Aero) as the prime contractor.

The JSF Program employs hundreds of models, simulations and support tools in its design and assessment activities. The data needs and outputs of these tools are tremendous, with significant overlaps and dependencies among the data produced and consumed by the tools. A glimpse into this complexity can be seen in Figure 1, which shows just some of the analytical data “food chain” across models and simulations. The types of data and sources of data cover a wide range. Data must be obtained from external authoritative sources (e.g., for threat data and environmental data) as well as produced internally (for JSF product data). The formats of data cover the gamut from structured files of text, to complex binary data, to relational databases.

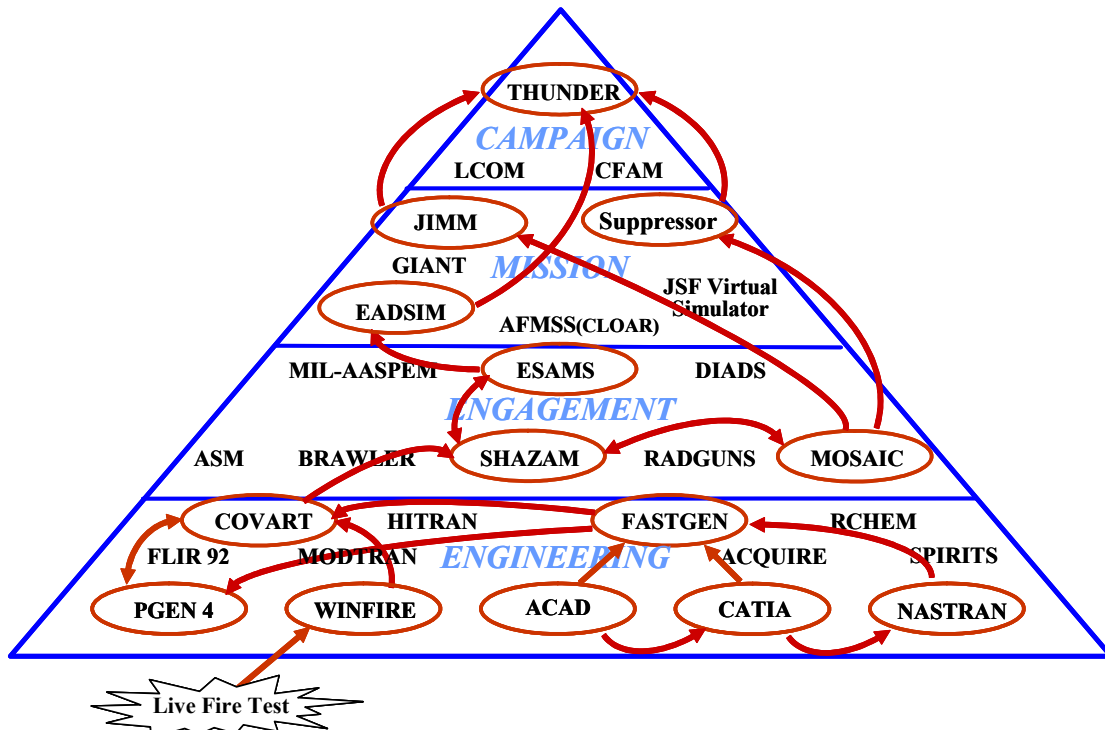


Figure 1. A Vulnerability Data Example of Data Dependencies Across Models

Ensuring that industry and government users of the JSF Suite of Models and Simulations, both the Engineering and Manufacturing Collaborative Environment (EMCE) and Strike Warfare Collaborative Environment (SWCE) [8] tools, have timely, secure access to the coherent, authoritative information needed to effectively perform their assigned tasks is the primary concern of the Modeling Information Sources Team (MISAT). The majority of the authors of this paper are charter members of the MISAT. It became evident early in the program life that metadata was the key to effectively managing JSF data. In addition to the traditional concepts of metadata that address the identification and description of data, it was evident to us that we had to establish the lineage and pedigree of data. By lineage, we mean the production chain of data, from its source outside the JSF program, or creation inside the program, through its transformation into simulation inputs, and following the simulation outputs through the analytical “food chain” with all the subsequent transformations of data.

To manage JSF M&S data, the MISAT is leading the development of the JSF Authoritative Modeling Information System (JAMIS). Figure 2 shows the notional way in which data flows through the JAMIS and gives a further example of the breadth of data managed. A key feature of the JAMIS architecture is the Resource Access System (RAS). The RAS provides navigation and access to all types of JAMIS data and is driven by metadata, rather than “hardwired” to support a set information structure.

2. JAMIS Metamodel

The goal of the JSF MISAT was to establish a common metadata representation that could be used for the breadth of JAMIS information types. The desired product is a data model for metadata, more commonly known as a metamodel. This metamodel specifies the kinds of information required to support JSF development and sustainment. It will guide the implementation and use of the various JAMIS databases. This metamodel must be suitable for forms of data ranging from data in relational databases, to structured text files (such as those used for computer-aided design, and simulation initialization and output), and to other structured representations such as Extensible Markup Language (XML) files. Additionally, this metamodel must address not only definition of data types, but also metadata for individual instances of data of those types. To be of value to M&S activities, and especially to the VV&A process, the model must capture the lineage (or pedigree) of data. The scope of the metamodel extends beyond what some may perceive as the traditional definition of metadata, which typically focuses on just the metadata for individual data resources. However, we feel that the metamodel must provide a complete, consistent coverage of the traditional metadata and the extended metadata necessary to support M&S use of data.

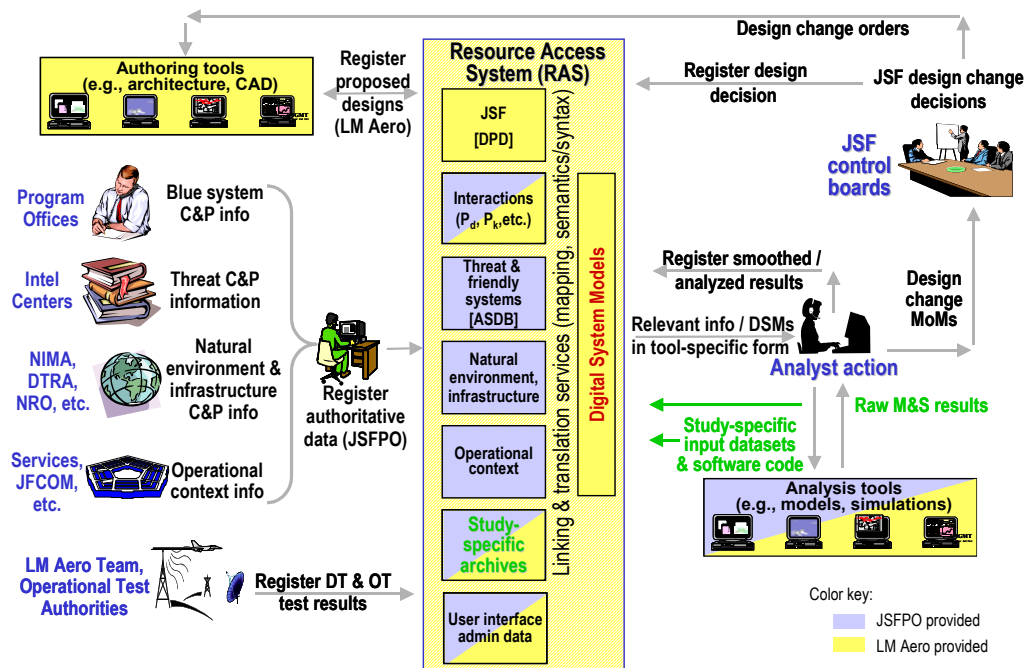


Figure 2. JAMIS Information Flow Concept

2.1 Existing Metadata Standards

Our goal was to use existing metadata standards where they were available and to only establish new metadata representations where no standard existed. We found that standards were plentiful for describing data types and even describing instance datasets. The primary dataset-level metadata standards incorporated into the JAMIS metamodel are the Dublin Core Standard [9] and the DoD Discovery Metadata Standard [10] (which draws from the Dublin Core and extends it substantially). In the area of VV&A, we made extensive use of the Data Quality Templates from the DoD VV&A Recommended Practices Guide [11]. However, when it came to metadata to describe the lineage of data, we found that there was a lack of standards and recommended practices. The elements of the JAMIS metadata model in that area represent the collective work of the JSF MISAT.

2.2 Representing the JAMIS Metamodel

While this paper describes the JAMIS metamodel in textual terms (because of space limitation), a more rigorous specification was needed to foster clear communication of concepts to implementers and users of the metamodel. In the tradeoffs of precision and unambiguous representation, ease of understandability, and transition from representation to implementation, we settled on a set of representations, rather than a

single one. To completely define all of the elements of the JAMIS metamodel and the associated relationships among the elements of the metamodel, we chose the IDEF1X entity-relationship representation. While this form of specification is extremely rigorous, it is not commonly understood by the entire audience for the metamodel, most specifically a substantial number of end users. To support this community, we also prepared an Excel spreadsheet listing the elements of the metamodel. Finally, to support the interchange of metadata, we established an XML Schema representation of the metamodel.

2.3 Major Elements and Relationships of the JAMIS Metamodel

In the following paragraphs, we will describe the major metamodel elements and the relationships of those elements. Figure 3 shows those major elements and their relationships. In that figure, a dot appears on the end of the relationship where multiple elements of the type may occur for any given element on the other end. To those familiar with relational terminology, this can be viewed as the entity-level description of the metamodel. In a few cases, we will describe the lowest-level data elements of the metamodel. Not all elements of the metamodel or relationships are covered in this paper, but the most critical ones are included.

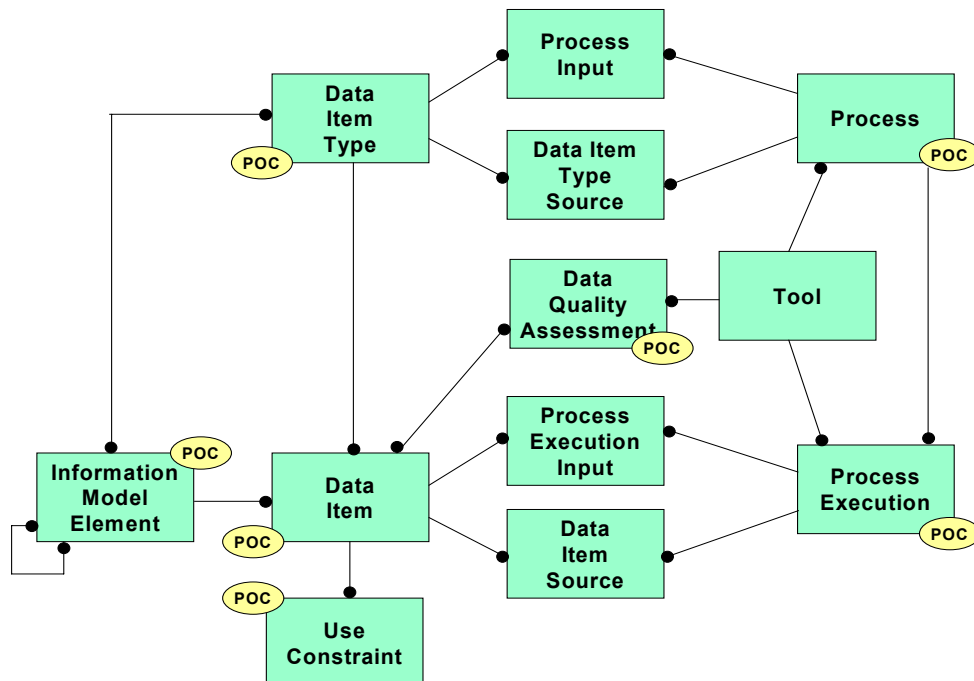


Figure 3. Major Elements of the JAMIS Metamodel

2.3.1 Core Elements

The primary elements of the metamodel are data item, data item type, and information model element. A data item type describes a type of data, of which there will potentially be multiple instances, i.e., data items. The data item type is the semantic description that is used for machine interpretation and processing. Data item types specify constraints that a data item must satisfy to be valid. The data item types needed for JAMIS data are complex. They consist of structured data such as tables, hierarchical structures, and portions of relational databases. For example, one of the radar cross section data item types specifies a table of specific dimensions, with specific named rows and specific units of measure.

Data items are the individually managed instances of data types and their associated metadata. For example, a type of simulation input file, e.g., a Brawler aircraft file, would be another data item type. The data items associated with this data item type would be the individual Brawler aircraft files (instances) managed in the JAMIS. In this example, a data item represents an entire dataset, but a data item could just as easily be used to represent some “slice” (some portion of one or more tables) in a relational database. In this abstract sense, a data item is created where a common set of metadata exists. This is a key concept in the JAMIS metamodel which allows us to apply the metamodel to the management of complete datasets, as well as portions of datasets (down to individual parameter values if appropriate).

Each data item has its own unique metadata. In addition to a unique identifier, this metadata includes such attributes as the name, version number, revision date, storage location, origin and access constraints associated with the data item. Most importantly, metadata is used to convey the meaning of the data, including the context in which it is valid. Thus metadata serves as the critical means by which data is managed and its appropriate use understood.

The information model element is a node in a data navigation tree (or set of trees). The set of relationships among information model elements defines the information model for the JAMIS data. The information model is used by the RAS as a card catalog to locate, access, and deliver data to users. While this tree of information model elements does not necessarily dictate where data items are stored, it does provide a way of organizing them into hierarchies. Data items are located via the RAS based on the information model. The recursive relationship between

an information model element and itself is used to capture the parent-child relationship in the information model tree(s). In addition to the hierarchical relationship among information model elements, the metamodel also supports cross-linking elements in different portions of the information model tree(s).

As can be seen in Figure 3, the metamodel also establishes point of contact (POC) relationships to data items, data item types, and information model elements. For data items and data item types, these relationships include authorship, release authority, and ownership. The POC relationship with an information model element establishes who created the node in the information model. POCs can be either individuals or organizations.

2.3.2 Data Quality Metadata

The metamodel captures historical information about data quality assessments performed on individual data items (or sets of data items). The data quality assessment element of the metamodel was drawn from the VV&A Recommended Practices Guide. Note that the metamodel supports the VV&A concept that validation occurs within the context of a specific tool. “Tool” as we will see later is an abstract concept used to describe any automated tool that uses or produces data, but in this case, it refers to the model or simulation for which this data quality assessment is applicable.

2.3.3 Use Constraint Metadata

The use constraint portion of the metamodel establishes those conditions under which use of the data item is appropriate. Types of use constraints include applicability over ranges of dates, to specific variants of a subject system (e.g., to only the carrier version of the JSF), or to a certain operational context.

It is particularly important for the JSF program to be able to communicate appropriate data use by identifying the operational contexts in which that data is valid. This is because JSF requirements are largely performance-based, and performance values are usually conditional, i.e., dependent on the operational environment and manner in which the JSF is used. As may be inferred from Figure 1, a data item which is the end result of a series of M&S-based analyses will have numerous operational context assumptions ‘built-in.’ The operational context portion of the metamodel allows these assumptions to be identified. By examining this metadata (via RAS), potential users of a data item will be able to determine its validity for the operational context they have in mind.

2.3.4 Data Item and Data Item Type Lineage

The metamodel establishes the lineage of data items through their relationship with process executions. The process execution input relationship is used to capture where one or more data items are used in a process execution. In most cases, the metamodel will be populated with data about process executions that represent model or simulation runs. In this case, the tool associated with the process execution is a specific model or simulation version. However, process execution can also be used to capture information about other processes, such as data preparation from external authoritative sources or data analysis to produce final results. The data item source relationship establishes what process execution created a data item. For example, if a simulation execution used three simulation input files and produced two output files, the inputs would be recorded as process execution inputs and the two outputs would be recorded as data item sources.

One can build up a tree of data production lineage by populating a JAMIS database (built in conformance with the metamodel) with the source and input relationships. Since the outputs of one model execution (process execution) are often the input to another model, the relationship builds up over several steps. This information becomes extremely powerful over time when inputs change. The metadata records can be examined to determine the impact of those changes, identifying where model executions should potentially be re-executed sequentially to produce new end results. Figure 4 shows an example of a data item lineage tree that would be captured using the metamodel. In this example, we can see that a change in data item DI_1 has an impact that can be easily determined, and the appropriate actions can be taken—that process executions PE_1 and PE_3 potentially need to be rerun to reproduce data items DI_5 and DI_8 .

This example also illustrates how the metamodel is key to addressing data coherency issues. In Figure 4 we see that data DI_3 is used in processes PE_1 and PE_2 . Thus, we must make sure that the correct data item is used in both processes. If inconsistent versions of DI_3 are used for the two processes, the result is data items produced higher in the production tree that are not coherent. In this example, the validity of item DI_8 is at stake. A concrete example would be to consider where CAD information is used as an input to two analysis chains that eventually produce radar cross section and aerodynamic performance data. If different CAD information were used to produce these two data items, a metamodel-based RAS query would raise a coherency warning flag. Absent further examination

(which might show the differences between the two CAD file versions are not significant), these two data items should not be used together to represent the aircraft.

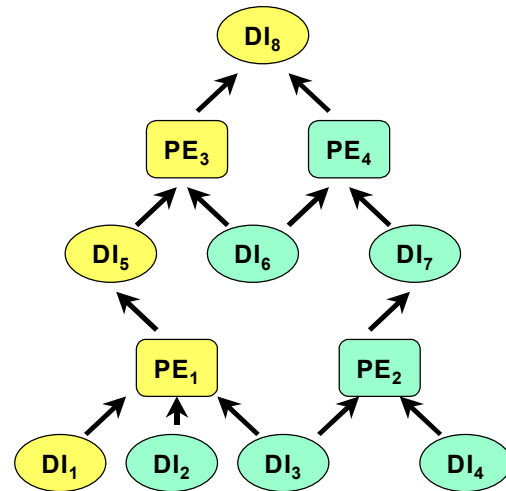


Figure 4. Data Item Lineage Example

Where process execution elements of the metamodel addresses individual data items, the same type of metadata can be captured about data item types. This is shown in the relationship between data item types and processes. This establishes the “normal” production process for data items of a given data item type. Once this data is populated, the metamodel user has a powerful quality control capability with which to gauge the completeness of metadata for individual data items. If one knows that a given process uses one set of data item types as inputs and another as outputs, the metadata for individual data items of those types and individual process executions can be examined and important questions can be addressed:

- If a process execution has been recorded, have all the inputs and outputs been registered?
- If a data item has been registered (which is of a data item type produced by some process) have the process execution and all the inputs been registered?

This metamodel can also be used to prompt the user for the properly complete set of metadata when new entries are made. For example, when a process execution is entered, the user can be prompted to enter metadata about the appropriate input and output data items, based on their process-data item type relationships.

3. Metamodel Implementations

In an earlier section, we described the breadth of data to be managed in the JAMIS and the shared responsibility for managing this data and making it available through a common interface. In the following sections, we will describe how the JAMIS metamodel is being implemented within LM Aero, to manage the JSF product data, and within Paradigm Technologies, Inc., to manage the characteristics and performance (C&P) data for threat systems and other, non-threat systems with which the JSF will interoperate. Through these discussions, we will demonstrate how the metamodel can be successfully applied to manage a broad range of data types, from individual datasets to the contents within a relational database.

3.1 Managing JSF Product Data

LM Aero, as the prime contractor, is responsible for the management of a wide variety of JSF product data. LM Aero's objective is to provide the entire JSF team timely, secure access to coherent, trustworthy product information for the JSF Air System. The LM Aero JSF team plans to use the commercial Metaphase product data management (PDM) system as the configuration management (CM) tool for the majority of product data. PDM systems do capture metadata as part of the process of authoring and registering data with an information system. However, to date Metaphase manages only a small portion of the product data required for the JSF Distributed Product Description (DPD) and only a portion of the metadata. Product data is currently stored in Metaphase, LiveLink, Merant PVCS and other configuration management systems. Locations and networks separate these systems physically and logically.

The RAS system will provide a uniform interface to the configuration-managed data within the JSF program. The interface will be consistent regardless of the storage location of the data and the system used to natively manage the data. Data will be organized in a product decomposition manner uniformly across configuration management systems. Additionally, users will be able to access the data with searches based on other metadata attributes such as and data item type or releasing organization.

With the JSF program using a federated PDM system, and RAS positioned in the architecture connecting to each of the individual PDM federates, three options exist for maintaining the metadata described in the JAMIS metamodel. The metadata can be managed in the individual PDM federates, it can be managed

within the RAS application, or the responsibility can be shared among these systems. The obvious advantage to storing and managing all the metadata in the RAS application is that it is a single point of implementation. The alternative of managing all the metadata in the PDM federates will require modifying multiple PDM systems. The current version of the RAS system implements the core of the metadata specification and alternative architectures are still being evaluated.

In any case, there are several requirements for the implementation of the JAMIS metadata model within the overall PDM architecture. Three critical concerns include performance, extensibility, and minimizing the burden of data entry on the creator of the data item.

3.1.1 JAMIS Performance Considerations

The implementation of the metamodel must be done in such a way that accessing data items using metadata attributes can be done in an efficient manner. It does not matter if the implementation is made in a relational or XML format; indexing the values of certain metadata attributes will be important.

3.1.2 Extensibility Considerations

It is not possible to foresee all of the metadata requirements for a thirty plus year program at the earliest stages of the program. With the changes in the tools and technology anticipated, it is very likely that entirely new data item types will come into existence as the program progresses. Additionally, given the wide scope of the engineering efforts involved, a complete survey of the metadata requirements of each group for each existing data item is not feasible. Changes to the metadata specification are inevitable and expected.

The implementation of the metadata model must be flexible enough to accept changes without forcing expensive code changes to the data definition language defining the data structures or to the application code that displays the data collection and presentation in the application.

The first step to achieving that flexibility is to work with the data at a higher level of abstraction than would be immediately expected. The instantiation of the model is done, in the language of the OMG Meta Object Facility Specification [12], at the metamodel level. If the implementation is in an XML format, the schema definition might be as simple as that which is shown in Figure 5. Translating the abstract recursive model to a more specific model for presentation and

data collection is a task handled through XML Stylesheet Language Transformation (XSLT) in the RAS application.

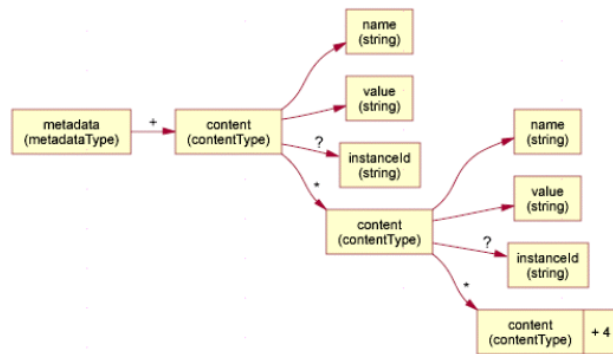


Figure 5. Abstract Metamodel Schema for XML Implementation

A similar abstraction can be created in a relational model with the metadata specification being implemented with a basic three-table structure as shown in Figure 6. By implementing with this more abstract approach, attributes and even entities can be added to the data model without changing the structure of the data repository or code used to present and collect the information.

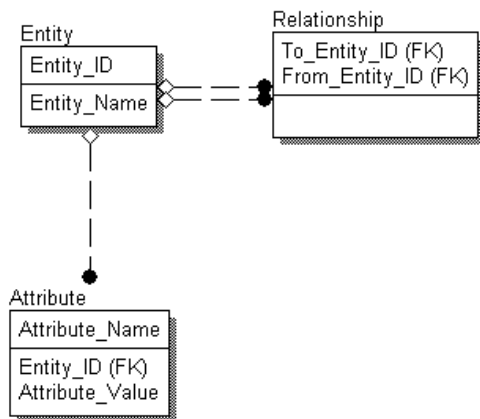


Figure 6. Abstract Metamodel Schema for Relational Implementation

3.1.3 Minimizing the Burden of Data Entry

The RAS and DPD teams are keenly aware that the enriched metamodel could place a burden on the individual uploading data to the system. The implementation of the metamodel needs to address that concern and minimize the impact on the system user. There are two basic approaches to minimizing that impact.

The first approach is to recognize that many entities found in the metamodel will not be implemented directly within the DPD since it would not be appropriate for the DPD to be the system of record for that information. A good example of this is the point of contact entities. The system of record for information about people remains in the human resources domain and it will feed the DPD implementation. Another example of this is the metamodel entity tool. The Virtual Enterprise System Configuration Management team manages the master tool list. By maintaining the details of these entities outside of the DPD, the user of the system will only need to select the identifying key for the element to provide the full scope of information about it.

A second step to minimizing the manual input of metadata is to automate the collection of the data by inferring some values from the uploading process. For example: the individual logged into the system and registering a data item can be inferred to be a POC for the data item. A data item of a certain type is produced by a set of known processes. These can be presented to the user as a 'pick list.' Just as a process takes input data items to create an output data item, a set of companion rules can be created taking the metadata from the data items serving as inputs and generating metadata for the outputs. Context of use constraints for a given data item may be inferred from the context of use constraints for the data items that served as inputs to the process creating the data item. A data item created in a process that uses two other data items as inputs, each with applicability to a different subset of aircraft, would logically have metadata reflecting an intersection of those subsets. In the same manner other rules can be created for specific metadata elements and specific processes.

Some metadata can be inferred from the data itself. Much of the data collected is in the form of complex data types such as arrays. In some cases, the range of an array implies a range value for a context of use.

3.1.4 The Challenge of Data Coherency

As engineers design a new product, they exploit models that simulate aspects of the product and its interactions with its external context. These models are generally software programs that abstract the physics of the real system or its environment. Executing these models for specific configurations of the product design and its context produces results files for subsequent analysis by the engineering team. Data results from one process become a part of the input for subsequent processes. Since the results depend almost completely on the particular input design configuration

and initialization parameters, it is critical to be able to trace the model execution results back to their ultimate source(s) and input specifications. Failing to do so in a complete, thorough manner opens the results up to misinterpretation, invalidation, and misuse. Recreating the results in the absence of complete configuration records becomes an expensive, hit-or-miss task. As we saw in the earlier description, the metamodel provides the lineage necessary to support the analysis of coherency.

3.2 Managing Threat and Friendly Systems Data

In this section we will see how the JAMIS metamodel is used to manage metadata about C&P data in the JSF Program Office's Authoritative Systems Database (ASDB). The ASDB is implemented as a series of M&S-specific data marts in a centralized Oracle 8i data warehouse.

The input data needs of the simulations of the JSF Strike Warfare Collaborative Environment are provided in a series of data marts. Each contains the C&P data of the JSF Program Office list of most important threat aircraft, air-to-air missiles, surface to air missile systems, anti-aircraft artillery, and weapons of the JSF. Data about other systems with which the JSF will interact will be added in the future.

These data marts were created from a variety of sources to include reverse engineering the data needs of the various M&S in the JSF toolkit, source databases from the intelligence agencies and ASDB-required structure and instance data marts. The ASDB is not only the collection of these data marts but also the back-end database loading tools and the front-end XML transformation tools that allow the user to extract the information he requires in tool-specific format with the most up-to-date information available that is documented by the JAMIS metadata.

To implement this design, the ASDB Team has created a Metadata Linking Model that will capture the pedigree at the instance level, shown in Figure 7. Based on an Oracle 8i platform, the ASDB data warehouse and the JAMIS metamodel database required a mechanism to link the two together at the attribute value level. The design the ASDB Team is employing links the data item in the metadata model with another entity in the ASDB Metadata Linking Database.

This design will capture every attribute value in an ASDB data mart as a record in the Data Item Source Map table, as indicated by the model above. In addition to instantiating this design, the ASDB team is writing triggers and stored procedures in Oracle to ensure

when the data mart is populated with authoritative data, a trigger will fire activating a stored procedure to populate other data marts and the Data Item Source Map table. Each ASDB Data Mart will have its own instantiation of this JAMIS metamodel and its own Metadata Linking Database.

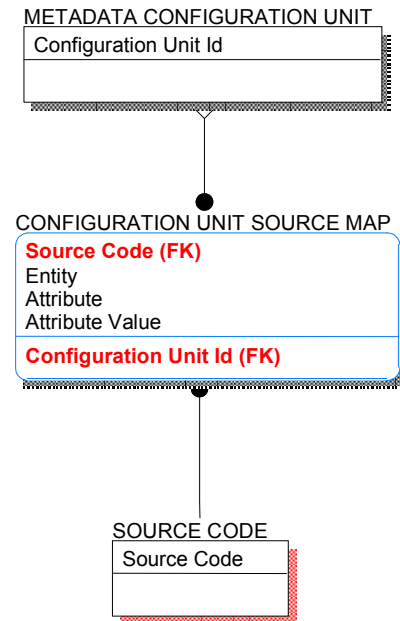


Figure 7. ASDB Metadata Linking Model

To give the reader an appreciation for the amount of data managed in the ASDB, the following example is provided. One of the tools supported by the ASDB is Thunder, a campaign-level simulation. In Thunder, there are many air-to-ground probabilities of kill (PKs), based on different sets of independent variables. The ASDB implementation of this metadata model will allow the customer to see the pedigree of every PK value in Thunder, currently over 750,000. This would be an overwhelming task to capture and maintain such a voluminous amount of metadata if it were not for the management technique this metamodel applies by capturing the metadata at the data item level. As discussed before, a data item can be a single attribute instance value, or in this case, one metadata record applies to 750,000 PK values. Only where the metadata is different is a new metadata record created.

4. Conclusions

The JAMIS metadata modeling efforts have resulted in a groundbreaking metamodel, which will increase user confidence in JAMIS data and support VV&A assessments. We are well on our way to implementing the metamodel, thus setting the gold standard for

metadata management in M&S. These metamodel concepts have been implemented in the first cycle of JAMIS deployment and were expanded in the second cycle. By relying on a metamodel-driven approach, the flexibility to change implementations in successive development cycles has been maintained. While the JAMIS implementation is specific to a single acquisition program, the metamodel has direct applicability to any application where the management of M&S data and the ability to record VV&A results are required.

5. References

- [1] J. Hollenbach, S. Bishop and W. Graves, "JSF Modeling Information Management," Simulation Interoperability Workshop, September 2002.
- [2] W. Graves, J. Hollenbach and M. Barnhart, "JSF Modeling Information Management," Simulation Interoperability Workshop, September 2002.
- [3] "A Roadmap for Simulation Based Acquisition – Report of the Joint Simulation Based Acquisition Task Force," Acquisition Council Draft for Coordination, 04 December 1998.
- [4] R. Frost and D. Thomen, "Simulation Based Acquisition, the Road Map," Simulation Interoperability Workshop, March 1999.
- [5] R. R. Lutz and J. F. Keane, "An Architecture for Simulation Based Acquisition," Simulation Interoperability Workshop, March 1999.
- [6] N. E. Karangelen, "The Simulation Based Acquisition Vision – A Brief Tutorial," proceedings of NDIA Workshop on Simulation Based Acquisition, Orlando, March 1998.
- [7] DoD Acquisition Council and SBA Industry Steering Group, Simulation Based Acquisition (SBA) Definition, 22 August 2000.
- [8] J. Coolahan, F.T. Case and R. Hartnett, "*JSF Strike Warfare Collaborative Environment (SWCE)*," Simulation Interoperability Workshop, September 2000.
- [9] National Information Standards Organization, "The Dublin Core Metadata Element Set," ANSI/NISO X39.85-2001, September 2001.
- [10] Deputy Assistant Secretary of Defense (Deputy Chief Information Officer), "Department

of Defense Discovery Metadata Standard," Preliminary Review Version 1.0, April 2003.

- [11] Defense Modeling and Simulation Office, "VV&A Recommended Practices Guide", Millennium Edition, <http://vva.dmsi.mil>.

- [12] Object Management Group, "Meta Object Facility Specification," Version 1.3, March 2000.

Author Biographies

ROY SCRUDDER is a program manager at the Applied Research Laboratories, Signal and Information Science Laboratory, The University of Texas at Austin. He holds a BS Degree in Mathematics from the University of Tennessee. Mr. Scrudder has over 25 years experience in systems analysis and design with the last 15 years working in support of various data engineering and data management projects in the defense community. He led the metadata modeling efforts for the JSF program.

W. HENSON GRAVES is a Technical Fellow at LM Aero and is the JAMIS Architect. He holds a BS degree in Mathematics from Tulane University in New Orleans and a Ph.D. in Mathematics from McMaster University in Hamilton, Ontario. He held a Post Doctoral Fellowship in Mathematical Physics at Queens University in Kingston, Ontario and was a Research Associate at Stanford University in Palo Alto. He has spent over 10 years in realizing the SBA vision at LM Space Systems and at LM Aero. Dr. Graves was Deputy Program Manager and technical lead at LM Space Systems for the Defense Advanced Project Agency (DARPA) Simulation Based Design Program and was formerly Professor of Computer Science at San Jose State University.

TOM TEIGEN holds degrees in Information Management Systems from Tarleton State University and Agricultural Engineering from the University of Minnesota. He has spent the last ten years designing and implementing database applications in a number of industries.

CHRIS JOHNSON is a Senior Staff Software Engineer at LM Aero. He holds BS and MS degrees in Mechanical Engineering from Virginia Polytechnic Institute and State University (Va. Tech) in Blacksburg, Virginia. He has spent 18 years at LM, first at Missiles & Fire Control in Orlando, Florida, and most recently at LM Aero in Fort Worth, Texas. He has worked on aircraft, rotorcraft, missile, avionics, and DARPA programs performing structural analyses and

developing tool and data integration frameworks. He has co-authored papers on multi-disciplinary optimization and served in developer and lead roles for design/analysis tools currently in use on aircraft programs at LM Aero.

STEVE HIX is the JSF ASDB project manager with Paradigm Technologies, Inc. He holds a BS in Economics from Rollins College in Winter Park, FL, an MBA from Golden Gate University in San Francisco, CA, and a Certificate in Systems Engineering from Old Dominion University in Norfolk, VA. Mr. Hix spent over 26 years in the aviation field as an air traffic controller in the Air Force and FAA and as a navigator and electronic warfare officer in a variety of Air Force aircraft including 12 combat missions in a B-52 during Desert Storm. The last three years he has been involved in database development in the M&S community.

JAMES W. HOLLENBACH holds a BS in Mechanical Engineering from Rutgers University, an MS in Aeronautical Systems from the University of West Florida, and is a graduate of the Program Manager Course at the Defense Systems Management College. A retired Navy captain, former EA-6B squadron commander and DoD Acquisition Professional, his last active duty assignment was as Director of the Defense Modeling and Simulation Office from 1994 to 1998, where he led development of the DoD and NATO M&S Master Plans and oversaw development of the High Level Architecture. His company, Simulation Strategies, Inc., provides modeling, simulation, and enterprise integration expertise to government and industry.